A practical framework and online tool for mutational signature analyses show inter-tissue variation and driver dependencies

Degasperi et al.

Supplementary Notes

Content:

1. Clustering with matching

1.a Problem description

- **Given:** n sets of vectors, each set has k vectors, for a total of kn vectors;
- **Objective:** cluster the kn vectors into k clusters such that the k vectors from the same set all belong to different clusters.

Think of this as: Let us say we have a problem that has no unique solution, and where a solution consists of a set of k vectors. We can run an algorithm many times and each time we obtain a set of k vectors as a solution. After n runs, we have n solutions, that is n sets of k vectors (Supplementary Fig. 1a). Clustering helps us find a consensus solution and tells us about robustness of the solutions.

1.b Algorithm Description

Intuition of how clustering with matching works: for each two sets, we can find a best match (i.e. a bijective function) pairing the k vectors in each set (e.g. based on least distance) and obtain k clusters with exactly two vectors in each cluster. We can then start grouping the matches starting from the best matches until we have clustered all the n sets.

Initialisation: compute a match between the vectors in one set and the ones in each of the other sets for a total of $n(n-1)/2$ matches. Each match pairs one vector from a set to one and only one vector on the other set based on shortest distance (Supplementary Fig. 1b). Algorithms for solving the assignment problem or stable

matching can be used to compute these matches. We use the assignment problem as this ensures that the match will be optimal in the sense that the average of the distances between all matched vectors will be the minimum possible.

After the $n(n-1)/2$ matches have been computed, we build a symmetric matrix A with the average distances obtained from each match $M_{i,j}$ (Supplementary Fig. 1c). Intuitively, the lower the average of the distances, the better the match is. This allows us to construct an ordered list of matches L, with $M_{i,j} \in L$, from best to worst (Supplementary Fig. 1d, Init. row).

Algorithm core: starting from the best match (lowest sum of distances in the match), group and merge matches until all vectors from all sets are clustered (Supplementary Fig. 1d, Step 3 row).

Assume there is:

- A matrix A of average distances obtained for the corresponding matches M
- an ordered list L of $n(n-1)/2$ matches from best to worst ($M_{i,j} \in L$)
- a pool G of grouped matches, initially empty ($G = \emptyset$)

Begin: remove the best match $M_{i,j}$ (the one for which $A_{i,j}$ is the lowest for all i and j) from L and add it to G
Step: remove the best match $M_{i,j}$ from L and

- if there are no matches with either i or j in G then add the match $M_{i,j}$ to G
- if only i (or j) but not j (or i) is in a match $M_{a,\dots,i,\dots,z}$ (or $M_{a,\dots,j,\dots,z}$) in G then combine $M_{i,j}$ with $M_{a,\dots,i,\dots,z}$ (or $M_{a,\dots,j,\dots,z}$), thus obtaining $M_{a,\dots,i,\dots,j,\dots,z}$. (this is a simple merge of tables on the index i, or j)
- if both i and j are in G then
    - if i and j are in the same match $M_{a,\dots,i,\dots,j,\dots,z}$ then do nothing (i and j are already merged)
    - if i and j are in different matches $M_{a1,\dots,i,\dots,z1}$ and $M_{a2,\dots,j,\dots,z2}$ then merge one match with $M_{i,j}$ and the result with the other match to obtain the match $M_{a1,a2,\dots,i,\dots,j,\dots,z1,z2}$

Stop: The algorithm stops when a single match with all n runs is obtained

1.c Example

Let us assume that after the initialisation step we obtain the matrix A and ordered list L (Supplementary Fig. 1d, Init. row). In this example, n=5 and k=3. Notice that the size of the vectors is only relevant for the computation of the distances between vectors, which is done at the initialisation step.

1. The first match in L, i.e. $M_{1,2}$, is the match with the lowest average distance of the match and thus it is added to G at the base step (Supplementary Fig. 1d, Base row);
2. At the following step, $M_{3,4}$ is also simply added to G (Supplementary Fig. 1d, Step 1 row), because neither run 3 nor 4 is already in G;
3. Then, $M_{2,3}$ is merged with $M_{1,2}$, because they share run 2 and there are no other matches in G with run 3, thus replacing $M_{1,2}$ with $M_{1,2,3}$ in G (Supplementary Fig. 1d, Step 2 row);
4. Finally, $M_{3,5}$ is merged with both $M_{1,2,3}$ and $M_{4,5}$, because it shares run 3 and run 5 with each of them respectively. This replaces $M_{1,2,3}$ and $M_{4,5}$ with $M_{1,2,3,4,5}$, which includes all 5 runs and terminates the algorithm (Supplementary Fig. 1d, Step 3 row).

1.d Time performance

To evaluate how the time performance of clustering with matching scales for increasing number of vectors (and thus clusters) k, in each set n, we prepared a series of test problems with n=100 and k=2,…,16 (Supplementary Fig. 2a). Each of the 15 problems is generated using (k-1) normal distributions, and data points are generated so that batches of k vectors are known to come from different, though unknown distributions.

We used an implementation in R of clustering with matching, which uses the R package *lpSolve* to solve the assignment problem. We also compared clustering with matching against constrained k-means, from the R package *conclust*.

Examples of clustering results for both clustering with matching and constrained k-means can be seen in Supplementary Fig. 2b for the problems with k=14, k=15 and k=16.

When considering the time required by the two algorithms to solve the problems using the considered implementations, it is clear that clustering with matching scales better than constrained k-means for increasing number of vectors k (Supplementary Fig. 2c).

2. Global signature extraction

We performed a single global extraction of substitution signatures by pooling 2,486 samples (excluding sig10 and sig7 hypermutated samples). The analysis of average silhouette width and reconstruction error suggested 24 signatures, illustrated in Extended Data Fig. 2.

First, there are signatures that are extracted well and clearly, and show high cosine similarities across all possible extraction exercises. However, if there were variations between tissues, these will not be detected. These signatures are (Extended Data Fig. 2): S18 (COSMIC1/Refsig1), S2 & S22 (COSMIC2/RefSig2 & COSMIC13/RefSig13), S20 (COSMIC11/RefSig11), S12 (COSMIC22/RefSig22), S24 (COSMIC16/RefSig16) and S13 (COSMIC19/RefSig19).

Second, there are signatures that resemble known signatures, but have a low cosine similarity with respect to known signatures or any of the organ specific signatures found in this study, which indicates a poor identification of these signatures. These signatures are: S7 (RefSig3/COSMIC3), S8 (RefSig4/COSMIC4), S5 (RefSig8/COSMIC8), S14 (RefSig9/COSMIC9)

Third, there are signatures where the global extract has caused ambiguity. For example, there seem to be several versions of MMR signatures (S6, S15, S16, S21), which may resemble organ specific signatures, but not necessarily any COSMIC signature or Reference Signatures (e.g. S15).

Fourth, there are signatures likely artefactual, bearing no similarity to any signature identified in this or previous studies: S1, S6, S9, S10, S17, S23.

While pooling together more samples could in principle improve the extraction of signatures shared across organs, in practice only few signatures are reliably obtained, while several artefactual signatures are also produced. The main reason for this is that that NMF extraction scales poorly with the number of signatures present in a set of samples (Alexandrov et. al 2012, Cell Reports). Given the relatively large number of signatures in the full dataset considered here (30-40), the extracted signatures are likely to be affected by the following issues:

1. Signatures that have flatter profiles are less likely to be obtained in an uncontaminated way
2. Strong signatures with particular, dominant peaks and also very high levels of mutagenesis associated with these patterns (e.g. mismatch repair-related signatures) tend to dominate "global" analyses and end up over-splitting driven by the minor variation between samples that are heavily mutated
3. Various efforts are often required such as excluding cohorts of samples and doing analyses separately for those highly mutated samples. In other words, there are pre-hoc adjustments and various post-hoc adjustments surrounding global analyses and fitting.

While some organ specific extractions may also present the above issues, others will not be affected, and the lower number of signatures in each individual organ will make addressing these issues more manageable.

3. SIGNAL: The Homepage of Mutational Signatures

Signal is an online, open-access mutational signature reference site built using modern web technologies. Signal is divided into two distinct sections: *Explore*, which allows users to explore cancer-derived and experimentally-generated signature data, and *Analyze*, which allows users to upload their own data for analysis by our pipeline (Supplementary Figure 3).

3.a Explore mutational signatures

The Explore section is organized into three categories corresponding to the source of mutagenesis—*in vivo* mutational signatures derived from human cancers (*Cancer*)[1-3]as described in the main text; *in vitro* mutational signatures derived from experiments involving environmental mutagens (*Environmental Mutagenesis*)[4] ; and *in vitro* mutational signatures derived from cell-based experiments involving CRISPR-Cas9 knockouts of different genes (*Gene-Edits*)[5].

The *Cancer* section presents the most up-to-date analyses of more than 3,000 whole-genome-sequenced (WGS) tumors across 21 different cancer types[2,3] as described in the main text. An interactive heatmap is the primary entry-point for accessing all cancer-derived mutational signature data (https://signal.mutationalsignatures.com/explore/cancer). Additionally, Signal provides an interactive signature network map (https://signal.mutationalsignatures.com/explore/cancer/network), allowing users to explore how signatures are related between organs. Users are also shown the similarities between *reference signatures* and the current set of 30 COSMIC mutational signatures[1].

The *Environmental Mutagenesis* section (https://signal.mutationalsignatures.com/explore/mutagens) is home to substitution and indel signatures (rearrangement numbers were insufficient to produce rearrangement signatures) derived from human induced pluripotent stem cells (iPSCs) that were exposed to 77 known or suspected environmental carcinogens (IARC Class I or IIa/IIb)[4]. Links to IARC classification, CAS number, and additional information on PubChem[6] are also provided. The *Gene-Edits* section https://signal.mutationalsignatures.com/explore/genes is home to substitution, rearrangement and indel signatures derived from a proof-of-principle study knocking out 9 genes related to DNA repair using CRISPR-Cas9 technology in human HAP-1 cells[5]. Information such as gene name, symbol, function, chromosomal location and known pathway(s) is provided for each gene with links to additional information on the NCBI

website (https://www.ncbi.nlm.nih.gov/gene). The user can view mutation profiles for each individual sample used in these studies.

3.b Analyze your data

The Analyze interface (https://signal.mutationalsignatures.com/analyse) allows users to upload somatic mutation data (aligned to either GRCh37 or GRCh38 currently) of a single sample, or set of samples, to perform quick mutational signature analysis. It is not designed to perform new mutational signature extractions. Rather, it assesses potential contributions of Signal's signatures to a sample's mutational profile, while being capable of also highlighting novel patterns.

As noted in the main manuscript, any given mutational profile can be theoretically modelled as a linear combination of mutational signatures. The larger the pool of mutational signatures used to derive a model, however, the more likely one is to discover, by chance, combinations not representative of the true biological history of a given sample (false positives). The accuracy of the signature fitting algorithm is therefore increased by selecting a biologically sensible subset of the available signatures for consideration in the model.

The user can simply specify the originating organ of their cancer sample and Signal will automatically select the appropriate candidate signatures on their behalf, leveraging the organ-specificity of the signatures in its database. Alternatively, Signal enables experienced users to manually select candidate signatures of their choice from the pool of signatures available on Signal (including cancer-derived and experimentally-derived signatures). We have made signatures from external sources available, namely COSMIC[7] and the Pan-Cancer Analysis Working Group on Mutational Signatures[8].

Signal returns the sample's mutational profile and estimated signature contributions, along with a reconstructed profile. Advanced users can interrogate data in more detail to understand the robustness (or otherwise) of their result, as facilitated by the bootstrap resampling process. Various additional analyses are offered including: the detection of transcriptional strand bias; the filtering of localised hypermutation (kataegis); and the identification of similar cancer samples in the database.

A note of caution: the fitting process is a purely mathematical procedure that will seek to fit whichever *a priori* signatures are selected. This does not mean that the biological process with which a given signature is associated is definitively present in the sample. Other means should be sought to formally confirm or refute the biological presence of a mutational process in any analysis.

Any difference between the original sample profile and the reconstructed profile may simply be noise that is unaccounted for. It could also be indicative of signatures that are present in the user's sample but were not selected for fitting, or were excluded by the sparsity filtering process. With this in mind, Signal may present the user with other potentially contributing signatures. If there is consistency in difference profiles between a subset or all of the user's samples, but with no similarity to any of the signatures in the database, this could indicate that a previously undiscovered signature is contributing to the sample's profile. The true biological process underpinning any unassigned or incorrectly assigned mutations would thus require further investigation. This specific functionality of our resource truly exploits the totality of knowledge that is present in this database.

Following a first release, this resource will be regularly updated, with the addition of more analyses, data and functionality. We welcome the contribution of experimental or cancer data by the community in order to enhance the database. We hope that Signal serves as a valuable resource to the community.

3.c Materials and methods

Signal is a modular website running in the flexible cloud environment, OpenStack[9]. Along with the tools Packer[10] and Terraform[11], OpenStack allows us to quickly and dependably deploy complex infrastructure. The Javascript frontend of Signal utilises React[12] and Redux[13] to create the modern user interface, while employing server-side rendering to ensure fast loading of the website. Browser storage (in the form of IndexedDB) is used to ensure the persistence of analysis results—which are not stored long-term on our servers—between sessions. Charts are powered by D3 using the Plotly[14] Javascript library. The Javascript frontend communicates with a RESTful API written using the Perl Dancer2[15] framework. These components are housed on multiple virtual machines within OpenStack and are served by a HAProxy[16] load balancer, allowing the website to be scaled quickly to the required demand.

Individual components of the analysis pipeline are written in Python and R and are managed by Workflow Runner[17] (WR). WR allocates resources to individual components of the pipeline—booting new OpenStack instances if necessary—and manages task dependencies. Distributed storage available to all instances in the environment is handled by Ceph[18]. Reference genomes are held in-memory on dedicated machines and served with a RESTful API to enable rapid lookups. All components are containerised using Docker.

3.d References

1 Alexandrov, L. B. *et al.* Signatures of mutational processes in human cancer. *Nature* **500**, 415-421, doi:10.1038/nature12477 (2013).

2 Campbell, P. J., Getz, G., Stuart, J. M., Korbel, J. O. & Stein, L. D. Pan-cancer analysis of whole genomes. *bioRxiv*, doi:10.1101/162784 (2017).

3 Nik-Zainal, S. *et al.* Landscape of somatic mutations in 560 breast cancer whole-genome sequences. *Nature* **534**, 47-54, doi:10.1038/nature17676 (2016).

4 Kucab, J. E. *et al.* A Compendium of Mutational Signatures of Environmental Agents. *Cell* **177**, 821-836 e816, doi:10.1016/j.cell.2019.03.001 (2019).

5 Zou, X. *et al.* Validating the concept of mutational signatures with isogenic cell models. *Nature communications* **9**, 1744, doi:10.1038/s41467-018-04052-8 (2018).

6 Kim, S. *et al.* PubChem 2019 update: improved access to chemical data. *Nucleic acids research* **47**, D1102-D1109, doi:10.1093/nar/gky1033 (2019).

7 Tate, J. G. *et al.* COSMIC: the Catalogue Of Somatic Mutations In Cancer. *Nucleic acids research* **47**, D941-D947, doi:10.1093/nar/gky1015 (2019).

8 Alexandrov, L. *et al.* The Repertoire of Mutational Signatures in Human Cancer. *bioRxiv*, doi:10.1101/322859 (2018).

9 *Openstack*, <https://www.openstack.org/software/> (2019).

10 *Packer*, <https://packer.io> (2019).

11 *Terraform*, <https://www.terraform.io> (2019).

12 *React*, <https://reactjs.org> (2019).

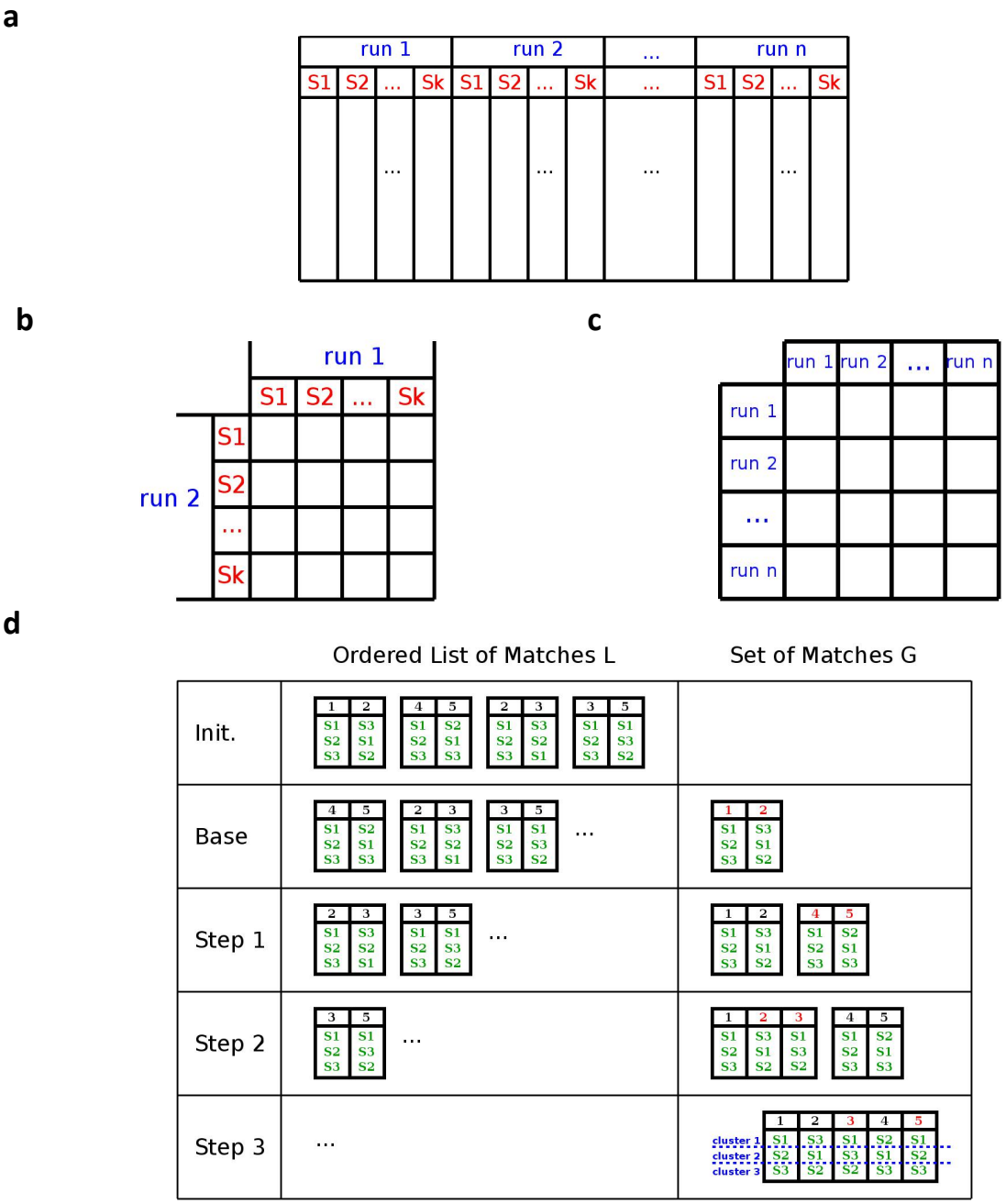
13 *Redux*, <https://redux.js.org> (2019).

14 *Plotly*, <https://plot.ly> (2019).

15 *PerlDancer*, <http://perldancer.org> (2019).
16 *HAProxy*, <http://www.haproxy.org> (2019).
17 *Workflow Runner*, <https://github.com/VertebrateResequencing/wr> (2019).
18 *Ceph*, <https://ceph.com> (2019).

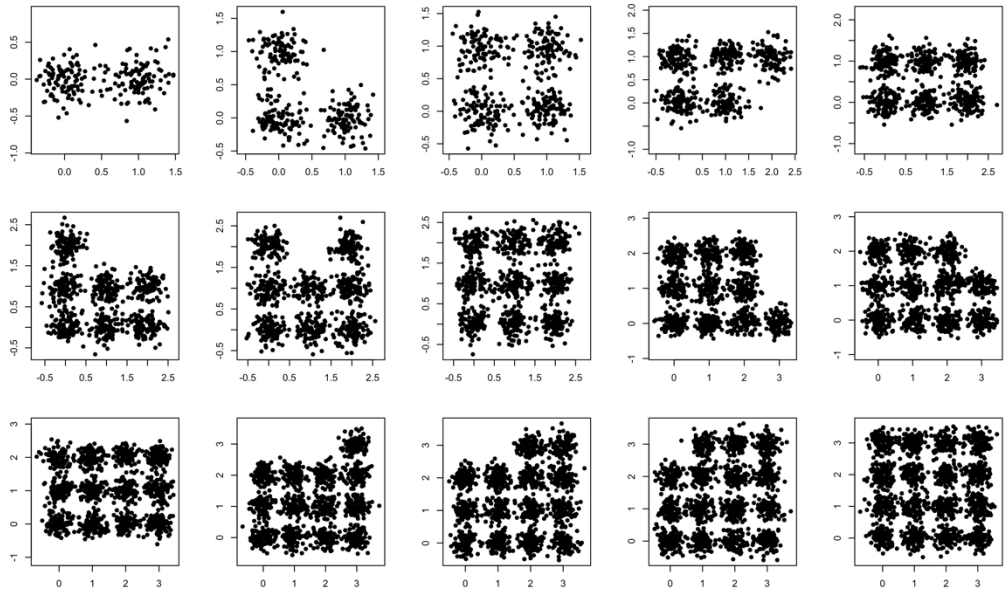
4. Supplementary Figures

Supplementary Figure 1. Clustering with matching. (a) Illustration of a matrix containing nk vectors to be clustered as columns. For each of n runs there are k vectors that should be clustered separately. **(b)** For each pair of runs, the distance between the k vectors in each run can be computed. This produces n(n-1)/2 distance matrices, and the figure illustrates one of them, i.e. the distance matrix for runs 1 and 2. **(c)** From each distance matrix in **(b)**, a match between the vectors in each run can be computed based on the lowest overall distance average of the match distances. The figure illustrates the n-by-n matrix A with the average for all the possible matches. **(d)** Illustration of the clustering with matching algorithm with n=5 and k=3. The initialisation procedure produces an ordered list of matches L (Init. row), starting from the match with the overall lowest average distance (M_1,2). Finally, matches can be merged sequentially in the G set until only one match table including all the n runs is obtained (from Base to Step 3 rows). Clusters are then obtained from the final merged match table (Step 3).

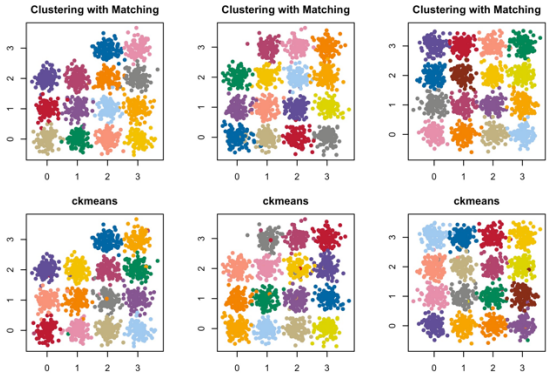


Supplementary Figure 2. Performance of Clustering with Matching and constrained k-means. Clustering problems for the time performance evaluation. Each of the 15 problems has number of runs n is 100, while the number of vectors (and clusters) k is increasing from 2 to 16. Data points are generated so that batches of k vectors are known to come from different, though unknown clusters. **(b)** Examples of clustering solutions to some of the problems in(a), obtained using clustering with matching (top row) and constrained k-means (bottom row). **(c)** Time required to solve the problems in (a), using either clustering with matching or constrained k-means.

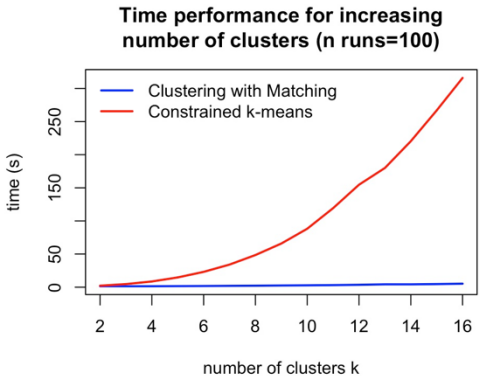
a



b



c



Supplementary Figure 3. Organization of the Signal web site. Two main sections are available: Explore and Analyze. In the Explore section, a database of mutational signatures of various types and from multiple sources can be explored interactively. In the Analyze section, users can upload mutational data and obtain estimates of mutational signatures activity.